

DIN EN 16602-80

ICS 35.080; 49.140

Einsprüche bis 2017-02-09

Entwurf

**Raumfahrtproduktsicherung –
Software-Produktsicherung;
Englische Fassung prEN 16602-80:2016**

Space product assurance –
Software product assurance;
English version prEN 16602-80:2016

Assurance produit des projets spatiaux –
Assurance produit logiciel;
Version anglaise prEN 16602-80:2016

Anwendungswarnvermerk

Dieser Norm-Entwurf mit Erscheinungsdatum 2016-12-09 wird der Öffentlichkeit zur Prüfung und Stellungnahme vorgelegt.

Weil die beabsichtigte Norm von der vorliegenden Fassung abweichen kann, ist die Anwendung dieses Entwurfs besonders zu vereinbaren.

Stellungnahmen werden erbeten

- vorzugsweise online im Norm-Entwurfs-Portal von DIN unter www.din.de/go/entwuerfe bzw. für Norm-Entwürfe der DKE auch im Norm-Entwurfs-Portal der DKE unter www.entwuerfe.normenbibliothek.de, sofern dort wiedergegeben;
- oder als Datei per E-Mail an nl@din.de möglichst in Form einer Tabelle. Die Vorlage dieser Tabelle kann im Internet unter www.din.de/go/stellungnahmen-norm-entwuerfe oder für Stellungnahmen zu Norm-Entwürfen der DKE unter www.dke.de/stellungnahme abgerufen werden;
- oder in Papierform an den DIN-Normenausschuss Luft- und Raumfahrt (NL), 10772 Berlin, Burggrafenstr. 6, 10787 Berlin.

Die Empfänger dieses Norm-Entwurfs werden gebeten, mit ihren Kommentaren jegliche relevanten Patentrechte, die sie kennen, mitzuteilen und unterstützende Dokumentationen zur Verfügung zu stellen.

Gesamtumfang 128 Seiten

DIN-Normenausschuss Luft- und Raumfahrt (NL)

Nationales Vorwort

Dieses Dokument (prEN 16602-80:2016) wurde vom Technischen Komitee CEN-CENELEC/TC 5 „Raumfahrt“ erarbeitet, dessen Sekretariat von DIN (Deutschland) gehalten wird.

Das zuständige deutsche Normungsgremium ist der Arbeitsausschuss NA 131-06-02 AA „Interoperabilität von Informations-, Kommunikations- und Navigationssystemen“ im DIN-Normenausschuss Luft- und Raumfahrt (NL).

Dieses Dokument (prEN 16602-80:2016) basiert auf ECSS-Q-ST-80C Rev.1 DIR1.

Dieses Dokument enthält unter Berücksichtigung des DIN-Präsidialbeschlusses 1/2004 nur die englische Originalfassung von prEN 16602-80:2016.

Dieses Dokument wurde speziell zur Behandlung von Raumfahrtsystemen erarbeitet und hat daher Vorrang vor jeglicher Europäischer Norm, die denselben Anwendungsbereich hat, jedoch über einen größeren Geltungsbereich (z. B. Luft- und Raumfahrt) verfügt.

Nationaler Anhang NA (informativ)

Begriffe und Abkürzungen

Die Benummerung der folgenden Begriffe und Abkürzungen ist identisch mit der Benummerung in der englischen Fassung.

3 Begriffe und Abkürzungen

3.1 Begriffe aus anderen Normen

Für die Anwendung dieses Dokuments gelten die Begriffe nach ECSS-ST-00-01 und insbesondere die folgenden Begriffe:

Abnahmeprüfung

Software-Produkt

ANMERKUNG Die Begriffe der Normen ECSS-E-ST-40 und ECSS-Q-ST-80 wurden einheitlich ausgelegt.

3.2 Für diese Norm spezifische Begriffe

3.2.1

automatische Code-Generierung

Generierung eines Quellcodes mit Hilfe eines Modelltools

3.2.2

Code-Abdeckung

prozentualer Anteil der Software, der beim Test ausgeführt (abgedeckt) wurde

3.2.3

Competent Assessor

Person, die die erforderlichen Fähigkeiten, Kompetenzen und Erfahrungen nachgewiesen hat, um ein Prozess-Assessment nach ISO/IEC 15504 durchzuführen

Anmerkung 1 zum Begriff: Aus ISO/IEC 15504:1998, Teil 9, übernommen.

3.2.4

Bedingung

boolescher Ausdruck, der keine booleschen Operatoren enthält

3.2.5

konfigurierbarer Code

Code (Quellcode oder ausführbarer Code), der durch das Einstellen von Parameterwerten angepasst werden kann

Anmerkung 1 zum Begriff: Diese Definition behandelt insbesondere Klassen konfigurierbarer Codes, die durch folgende Konfiguration erhalten wurden:

- Konfiguration, die auf der Anwendung einer Kompilations-Richtlinie basiert;
- Konfiguration, die auf der Anwendung einer Link-Richtlinie basiert;

- Konfiguration, die durch einen Parameter ausgeführt wurde, der in einer Konfigurationsdatei definiert ist;
- Konfiguration, die mit Daten ausgeführt wurde, die in einer Datenbank mit Einfluss auf die tatsächlich ausführbaren Softwareteile definiert sind (z.B. Parameter, die Branchenstrukturen definieren, die dazu führen, dass vorhandene Teile des Codes nicht ausgeführt werden).

3.2.6

COTS-, OTS-, MOTS-Software

für die Anwendung dieser Norm handelsübliche gebrauchsfähige Software, gebrauchsfähige Software und modifizierte Standardsoftware, für die ein Benutzungsnachweis verfügbar ist

3.2.7

kritische Software

Software der Kritikalitäts-Kategorie A, B oder C

Anmerkung 1 zum Begriff: Siehe ECSS-Q-ST-80C, Anhang D.1 – Kritikalitäts-Kategorien für Software.

3.2.8

deaktivierter Code

Code, der obwohl er durch korrektes Design und Codieren integriert wurde, nur in bestimmten Konfigurationen des Softwareprodukts oder in keiner Konfiguration ausgeführt werden soll

[aus RTCA/DO-178B übernommen]

3.2.9

Entscheidung

boolescher Ausdruck aus Bedingungen und Null oder mehreren Booleschen Operatoren, die in einem Kontrollkonstrukt verwendet werden

Anmerkung 1 zum Begriff: Beispielsweise: sind die Ausdrücke „if ... then ... else“ oder „case“ Kontrollkonstrukte.

Anmerkung 2 zum Begriff: Eine Entscheidung ohne booleschen Operator ist eine Bedingung.

Anmerkung 3 zum Begriff: Wenn eine Bedingung häufiger als einmal in einer Entscheidung erscheint, ist jedes Auftreten eine einzelne Bedingung.

3.2.10

Entscheidungsüberdeckung

Maßnahme des Programmteils, in dem jeder Punkt des Eingangs und Ausgangs mindestens einmal aufgerufen ist und jede Entscheidung mindestens einmal „Wahr“- und „Falsch“-Werte (en: „true“ and „false“ values) angenommen hat

Anmerkung 1 zum Begriff: Entscheidungsüberdeckung enthält laut Definition Aussageüberdeckung.

3.2.11

vorhandene Software

Software, die außerhalb der Geschäftsvereinbarung entwickelt wurde, für welche diese Norm gilt, einschließlich Software früherer Entwicklungen, die vom Lieferanten zur Verfügung gestellt wurden, Software von früheren Entwicklungen, die vom Kunden zur Verfügung gestellt wurden, COTS-, OTS- und MOTS-Software, Freeware und Open-Source-Software

3.2.12

Integrationstest

Test, bei dem Softwarekomponenten, Hardwarekomponenten oder beide kombiniert und getestet werden, um die Wechselwirkungen zwischen ihnen zu bewerten

[QUELLE: IEEE 610.12:1990]

3.2.13

logisches Modell

implementierungsunabhängiges Modell von Softwareprodukten für die Analyse und Dokumentation von Softwareanforderungen

3.2.14

Reservenphilosophie

Grundprinzip für das Festlegen von Reserven, die Leistungsparametern und Computerausrüstung in einer bestimmten Entwicklungsphase zugeordnet werden, und wie diese Reserven im Projektablauf genutzt werden

3.2.15

Metrik

definiertes Messverfahren und Größenwertskala

Anmerkung 1 zum Begriff: Metriken können intern oder extern sowie direkt oder indirekt sein.

Anmerkung 2 zum Begriff: Metriken umfassen Verfahren zur Kategorisierung qualitativer Daten.

[QUELLE: ISO/IEC 9126-1:2001]

3.2.16

Migration

Umstellung eines Softwareprodukts auf eine vollständig neue Umgebung

3.2.17

Missionsprodukte

Produkte und Dienstleistungen, die vom Raumfahrtsystem geliefert werden

Anmerkung 1 zum Begriff: Kommunikationsdienste, wissenschaftliche Daten

3.2.18

modifizierte Bedingungs- und Entscheidungsüberdeckung

Maßnahme des Programmteils, in dem jeder Punkt des Eingangs und Ausgangs mindestens einmal aufgerufen ist und jede Entscheidung mindestens einmal „Wahr“- und „Falsch“-Werte (True/False) angenommen hat und für jede Bedingung in einer Entscheidung nachgewiesen wurde, dass sie das Ergebnis dieser Entscheidung unabhängig beeinflusst hat

Anmerkung 1 zum Begriff: Für eine Bedingung wird nachgewiesen, dass sie das Ergebnis einer Entscheidung unabhängig beeinflusst, indem diese Bedingung verändert wird, während alle anderen möglichen Bedingungen unverändert bleiben.

3.2.19

betriebllich

für die Zwecke dieser Norm auf die Softwareanwendung bezogen

Anmerkung 1 zum Begriff: Bezieht sich nicht auf den Betrieb des Raumschiffs.

3.2.20

Übertragbarkeit (ein Qualitätsmerkmal)

Fähigkeit der Software, von einer Umgebung zu einer anderen Umgebung übertragen zu werden

3.2.21

Qualitätsmerkmale (Software)

Reihe von Attributen eines Softwareprodukts, durch die dessen Qualität beschrieben und beurteilt wird

Anmerkung 1 zum Begriff: Ein Softwarequalitätsmerkmal kann über mehrere Stufen in Teilmerkmale verfeinert werden.

3.2.22

Qualitätsmodell (Software)

Reihe von Merkmalen und die Beziehung zwischen ihnen, die die Grundlage für die Festlegung der Qualitätsanforderungen und der Qualität der Beurteilung liefern

[QUELLE: ISO/IEC 9126-1:2001]

3.2.23

Echtzeit

charakteristisch für ein(en) Betriebssystem oder -modus, das/der innerhalb der tatsächlichen Zeitspanne, in der ein externer Prozess stattfindet, Berechnungen durchführt, so dass die Rechenergebnisse zur Steuerung, Überwachung oder Reaktion nahezu simultan mit dem externen Prozess verwendet werden können

[QUELLE: IEEE 610.12:1990]

3.2.24

Regressionstest (Software)

selektives wiederholtes Testen eines Systems oder einer Komponente, um nachzuweisen, dass die Modifikationen keine unbeabsichtigten Auswirkungen verursacht haben und dass das System oder die Komponente weiterhin den festgelegten Anforderungen entspricht

[QUELLE: IEEE 610.12:1990]

3.2.25

Wiederverwendbarkeit

Umfang, in dem eine Softwareeinheit oder ein anderes Arbeitsprodukt in mehr als einem Computerprogramm oder Softwaresystem verwendet werden kann

[QUELLE: IEEE 610.12:1990]

3.2.26

Einzelvorgabe

Vorgabe für eine Singularität der Funktion

3.2.27

Software

siehe „Softwareprodukt“ in ECSS-S-ST-00-01

3.2.28

Softwarekomponente

Bestandteil eines Softwaresystems

Anmerkung 1 zum Begriff: Softwarekomponente wird als allgemeine Bezeichnung verwendet.

Anmerkung 2 zum Begriff: Softwarekomponenten sind zusammengesetzt und können zur Bildung neuer Komponenten zerlegt werden. In der Produktionsphase haben Komponenten die Form von Modulen, Aufgaben oder Programmen, von denen jede wiederum Konfigurationseinheiten sein können. Die Anwendung des Begriffs erfolgt allgemeiner als bei ANSI/IEEE, wo eine Komponente als ein „Grundbaustein eines Systems oder Programms“ definiert ist; nach dieser Norm sind Komponenten jedoch nicht immer „Grundbausteine“, da sie zerlegt werden können.

3.2.29

softwareintensives System

Raumfahrtssystem, das vorwiegend aus Softwareelementen besteht

Anmerkung 1 zum Begriff: In derartigen Systemen bestehen die Teilsysteme hauptsächlich aus Software. Bei dieser Art von Systemen sind die meisten Schnittstellen Software-Software-Schnittstellen.

3.2.30

Softwareeinheit

siehe „Softwareprodukt“ in ECSS-S-ST-00-01

3.2.31

Überprüfbarkeit der Software

Eigenschaft eines Systems, die erlaubt, dass die Werte der Zustandsvariablen während der Untersuchungen der Ergebnisvariablen bestimmt werden können

3.2.32

Softwareproblem

Zustand eines Softwareprodukts, der Schwierigkeiten oder Unsicherheiten bei der Verwendung der Software verursacht

[QUELLE: CMU/SEI-92-TR-022]

3.2.33

Softwareproduktsicherung

Gesamtheit der Aktivitäten, Normen, Steuerungen und Verfahren in der Lebensdauer eines Softwareprodukts, die Vertrauen erzeugt, dass das gelieferte Softwareprodukt oder Software, die die Qualität des gelieferten Produkts beeinflusst, den Anforderungen des Kunden entspricht

3.2.34

Softwareelement

einzel kompilierbarer Teil eines Quellcodes

Anmerkung 1 zum Begriff In dieser Norm wird nicht zwischen einem Softwareelement und einer Datenbank unterschieden; für beide gelten die gleichen Anforderungen.

3.2.35

Aussageüberdeckung

Maßnahme des Programmteils, innerhalb dessen jede durchführbare Aussage für den Quellcode mindestens einmal ausgeführt wurde

3.2.36

Stresstest

Test, mit dem ein System oder eine Softwarekomponente bis an die Grenzen der dafür festgelegten Anforderungen oder darüber hinaus beansprucht wird

3.2.37

Testfall

Reihe von Testvorgaben, Testbedingungen und erwarteten Ergebnissen, die für ein bestimmtes Ziel entwickelt wurden, z. B. um einen bestimmten Programmpfad zu testen oder Übereinstimmung mit einer spezifischen Anforderung nachzuweisen

3.2.38

Testdesign

Dokument, das die Einzelheiten des Testverfahrens für eine Softwareeigenschaft oder eine Kombination von Softwareeigenschaften festlegt und die dazugehörigen Tests identifiziert

3.2.39

Testverfahren

detaillierte Anweisungen zum Testaufbau, zur Durchführung und zur Beurteilung der Ergebnisse für einen bestimmten Test

3.2.40

Testskript

Datei, die eine Reihe von in einem gängigen Format geschriebene (rechner- oder toolgestützte) Befehle oder Anweisungen enthält, um die Durchführung eines Testverfahrens oder einer Kombination von Testverfahren (und die damit verbunden Auswertung der Ergebnisse) zu automatisieren

3.2.41

Elementtest

Test eines einzelnen Softwareelements

3.2.42

nicht erreichbarer Code

Code, der aufgrund des Designs oder eines Codierungsfehlers nicht ausgeführt werden kann

3.2.43

Gebrauchstauglichkeit (ein Qualitätsmerkmal)

Fähigkeit der Software, vom Anwender verstanden, gelernt, genutzt und geschätzt zu werden, wenn sie unter bestimmten Bedingungen angewendet wird

3.2.44

Validierung

<Software> Prozess zur Bestätigung, dass die Grundanforderungen für Funktion und Leistung des Endprodukts korrekt und vollständig erfüllt sind

3.2.45

Verifizierung

<Software> Prozess zur Bestätigung, dass für jede Aktivität ausreichend Spezifikationen und Vorgaben vorhanden sind und dass die Ergebnisse korrekt sind und mit den Spezifikationen und Vorgaben übereinstimmen

3.2.46

Walk-through

Durchlauf

statische Analysetechnik, bei der ein Designer oder Programmierer die Mitglieder eines Entwicklungsteams und andere interessierte Parteien durch ein Softwareprodukt führt und die Teilnehmer bittet, Fragen zu stellen und sich zu möglichen Fehlern, Verstößen gegen Entwicklungsnormen und zu weiteren Problemen zu äußern

[QUELLE: IEEE 1028-1997]

3.3 Abkürzungen

Für die Anwendung dieses Dokuments und ECSS-E-ST-40 gelten die Abkürzungen nach ECSS-S-ST-00-01 und die folgenden Abkürzungen:

Zur Definition der DRD-Abkürzungen, siehe Anhang A.

ANMERKUNG Die Abkürzungen der Normen ECSS-E-ST-40 und ECSS-Q-ST-80 wurden einheitlich ausgelegt.

Abkürzung	Bedeutung	
------------------	------------------	--

AR	Abnahmebewertung	(en: acceptance review)
	ANMERKUNG Der Begriff SW-AR kann verwendet werden, um herauszustellen, dass es sich ausschließlich um Abnahmebewertungen in Bezug auf Software-Produkte handelt.	
CDR	kritische Entwurfsprüfung	(en: critical design review)
	ANMERKUNG Der Begriff SW-CDR kann verwendet werden, um herauszustellen, dass es sich ausschließlich um kritische Entwurfsprüfungen in Bezug auf Software-Produkte handelt.	
CMMI	Prozessmodell zur Beurteilung und Verbesserung der Qualität („Reife“) von Produkt-Entwicklungsprozessen	(en: capability maturity model integration)
COTS	handelsübliches Produkt	(en: commercial off-the-shelf)
CPU	zentrale Prozessoreinheit	(en: central processing unit)
DDF	Datei zur Festlegung der Konstruktion	(en: design definition file)
DDR	detaillierte Entwurfsprüfung	(en: detailed design review)
DJF	Datei zur Begründung der Konstruktion	(en: design justification file)
DRD	Definition der Dokumentanforderungen	(en: document requirements definition)
ECSS	Europäische Kooperation für Raumfahrtnormung	(en: European Cooperation for Space Standardization)
eo	erwartetes Ergebnis	(en: expected output)
GS	Bodensegment	(en: ground segment)
HMI	Mensch-Maschine-Schnittstelle	(en: human machine interface)
HSIA	Hardware/Software-Interaktionsanalyse	(en: hardware-software interaction analysis)
HW	Hardware	
ICD	Schnittstellenidentifikationsdokument	(en: interface control document)
INTRSA	Internationales Registrierungsschema für Assessoren	(en: international registration scheme for assessors)
IRD	Schnittstellenanforderungsdokument	(en: interface requirements document)

Abkürzung	Bedeutung	
ISO	Internationale Organisation für Normung	(en: International Organization for Standardization)
ISV	unabhängige Softwarevalidierung	(en: independent software validation)
ISVV	unabhängige Softwareverifizierung und -validierung	(en: independent software verification and validation)
MGT	Managementdatei	(en: management file)
MF	Instandhaltungsdatei	(en: maintenance file)
MOTS	serienmäßig produziertes Produkt mit möglichen Modifikationen bzw. Anpassungen nach Erwerb	(en: modified off-the-shelf)
OBCP	bordseitige Kontrollprozedur	(en: on-board control procedure)
OP	Betriebsplan	(en: operational plan)
ORR	Bewertung der Einsatzbereitschaft	(en: operational readiness review)
OTS	serienmäßig produziert	(en: off-the-shelf)
PAF	Produktsicherungsdatei	(en: product assurance file)
PDR	vorläufige Entwurfsprüfung	(en: preliminary design review)
	ANMERKUNG Der Begriff SW-PDR kann verwendet werden, um herauszustellen, dass es sich ausschließlich um vorläufige Entwurfsprüfungen in Bezug auf Software-Produkte handelt.	
PRR	vorläufige Anforderungsbewertung	(en: preliminary requirement review)
QR	Qualifikationsbewertung	(en: qualification review)
	ANMERKUNG Der Begriff SW-QR kann verwendet werden, um herauszustellen, dass es sich ausschließlich um Qualifikationsbewertungen in Bezug auf Software-Produkte handelt.	
RB	Grundanforderungen	(en: requirements baseline)
SCAMPI	genormtes CMMI-Appraisal-Verfahren zur Prozessverbesserung	(en: standard CMMI appraisal method for process improvement)
SDE	Softwareentwicklungsumgebung	(en: software development environment)
SOS	Softwarebetriebsunterstützung	(en: software operation support)
SPA	Softwareproduktsicherung	(en: software product assurance)
SPAMR	Meilenstein-Bericht zur	(en: software product assurance milestone)

Abkürzung	Bedeutung	
	Softwareproduktsicherung	report)
SPAP	Plan der Softwareproduktsicherung	(en: software product assurance plan)
SPR	Softwareproblembericht	(en: software problem report)
SRB	Prüfungsausschuss für Software	(en: software review board)
SRR	Bewertung der Systemanforderungen	(en: system requirements review)
	ANMERKUNG Der Begriff SW-SRR kann verwendet werden, um herauszustellen, dass es sich ausschließlich um Bewertungen der Systemanforderungen in Bezug auf Software-Produkte handelt.	
SW	Software	(en: software)
SWE	Software-Engineering	(en: software engineering)
TRR	Bewertung der Prüfungsbereitschaft	(en: test readiness review)
TS	technische Spezifikation	(en: technical specification)

— Leerseite —

CEN/CLC/TC 5

Date: 2016-4

prEN 16602-80:2016

CEN/CLC/TC 5

Secretariat: DIN

Space product assurance - Software product assurance

Raumfahrtproduktsicherung - Software-Produktsicherung

Assurance produit des projets spatiaux - Assurance produit logiciel

ICS:

Descriptors:

Document type: European Standard

Document subtype:

Document stage: CEN ENQUIRY

Document language: E

Table of contents

Foreword	8
1 Scope.....	9
2 Normative references	10
3 Terms, definitions and abbreviated terms.....	11
3.1 Terms for other standards.....	11
3.2 Terms specific to the present standard	11
3.3 Abbreviated terms.....	16
4 Space system software product assurance principles	19
4.1 Introduction.....	19
4.2 Organization of this Standard	20
4.3 Tailoring of this Standard	22
5 Software product assurance programme implementation	23
5.1 Organization and responsibility	23
5.1.1 Organization.....	23
5.1.2 Responsibility and authority	23
5.1.3 Resources.....	24
5.1.4 Software product assurance manager/engineer	24
5.1.5 Training.....	24
5.2 Software product assurance programme management.....	25
5.2.1 Software product assurance planning and control.....	25
5.2.2 Software product assurance reporting.....	26
5.2.3 Audits.....	27
5.2.4 Alerts.....	27
5.2.5 Software problems	27
5.2.6 Nonconformances.....	28
5.2.7 Quality requirements and quality models.....	28
5.3 Risk management and critical item control.....	29
5.3.1 Risk management	29

5.3.2	Critical item control.....	29
5.4	Supplier selection and control.....	29
5.4.1	Supplier selection.....	29
5.4.2	Supplier requirements.....	30
5.4.3	Supplier monitoring.....	30
5.4.4	Criticality classification.....	31
5.5	Procurement.....	31
5.5.1	Procurement documents.....	31
5.5.2	Review of procured software component list.....	31
5.5.3	Procurement details.....	31
5.5.4	Identification.....	31
5.5.5	Inspection.....	32
5.5.6	Exportability.....	32
5.6	Tools and supporting environment.....	32
5.6.1	Methods and tools.....	32
5.6.2	Development environment selection.....	33
5.7	Assessment and improvement process.....	33
5.7.1	Process assessment.....	33
5.7.2	Assessment process.....	34
5.7.3	Process improvement.....	35
6	Software process assurance.....	36
6.1	Software development life cycle.....	36
6.1.1	Life cycle definition.....	36
6.1.2	Process quality objectives.....	36
6.1.3	Life cycle definition review.....	36
6.1.4	Life cycle resources.....	36
6.1.5	Software validation process schedule.....	37
6.2	Requirements applicable to all software engineering processes.....	37
6.2.1	Documentation of processes.....	37
6.2.2	Software dependability and safety.....	38
6.2.3	Handling of critical software.....	40
6.2.4	Software configuration management.....	42
6.2.5	Process metrics.....	44
6.2.6	Verification.....	45
6.2.7	Reuse of existing software.....	48
6.2.8	Automatic code generation.....	51

6.3	Requirements applicable to individual software engineering processes or activities.....	52
6.3.1	Software related system requirements process.....	52
6.3.2	Software requirements analysis	52
6.3.3	Software architectural design and design of software items	54
6.3.4	Coding	55
6.3.5	Testing and validation	56
6.3.6	Software delivery and acceptance.....	61
6.3.7	Operations	62
6.3.8	Maintenance	63
7	Software product quality assurance	65
7.1	Product quality objectives and metrication	65
7.1.1	Deriving of requirements	65
7.1.2	Quantitative definition of quality requirements	65
7.1.3	Assurance activities for product quality requirements	65
7.1.4	Product metrics	65
7.1.5	Basic metrics.....	66
7.1.6	Reporting of metrics	66
7.1.7	Numerical accuracy.....	66
7.1.8	Analysis of software maturity.....	67
7.2	Product quality requirements	67
7.2.1	Requirements baseline and technical specification	67
7.2.2	Design and related documentation.....	68
7.2.3	Test and validation documentation.....	68
7.3	Software intended for reuse	69
7.3.1	Customer requirements.....	69
7.3.2	Separate documentation	69
7.3.3	Self-contained information.....	69
7.3.4	Requirements for intended reuse	69
7.3.5	Configuration management for intended reuse.....	69
7.3.6	Testing on different platforms.....	70
7.3.7	Certificate of conformance	70
7.4	Standard ground hardware and services for operational system.....	70
7.4.1	Hardware procurement	70
7.4.2	Service procurement.....	70
7.4.3	Constraints.....	71
7.4.4	Selection	71

7.4.5	Maintenance	71
7.5	Firmware	71
7.5.1	Device programming	71
7.5.2	Marking	72
7.5.3	Calibration	72
Annex A (informative) Software documentation.....		73
Annex B (normative) Software product assurance plan (SPAP) - DRD		79
B.1	DRD identification	79
B.1.1	Requirement identification and source document	79
B.1.2	Purpose and objective	80
B.2	Expected response	81
B.2.1	Scope and content	81
B.2.2	Special remarks	85
Annex C (normative) Software product assurance milestone report (SPAMR) - DRD		86
C.1	DRD identification	86
C.1.1	Requirement identification and source document	86
C.1.2	Purpose and objective	86
C.2	Expected response	87
C.2.1	Scope and content	87
C.2.2	Special remarks	88
Annex D (normative) Tailoring of this Standard based on software criticality.....		89
D.1	Software criticality categories	89
D.2	Applicability matrix	90
Annex E (informative) List of requirements with built-in tailoring capability.....		101
Annex F (informative) Document organization and content at each milestone.....		102
F.1	Introduction	102
F.2	ECSS-Q-ST-80 Expected Output at SRR	102
F.3	ECSS-Q-ST-80 Expected Output at PDR	104
F.4	ECSS-Q-ST-80 Expected Output at CDR	109
F.5	ECSS-Q-ST-80 Expected Output at QR	111
F.6	ECSS-Q-ST-80 Expected Output at AR	112

F.7 ECSS-Q-ST-80 Expected Output not associated with any specific milestone review	114
--	-----

Bibliography.....	116
--------------------------	------------

Figures

Figure 4-1: Software related processes in ECSS Standards.....	20
Figure 4-2: Structure of this Standard.....	21
Figure A-1 : Overview of software documents	73

Tables

Table A-1 : ECSS-E-ST-40 and ECSS-Q-ST-80 Document requirements list (DRL)	74
Table B-1 : SPAP traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses	79
Table C-1 : SPAMR traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses	86
Table D-1 : Software criticality categories.....	89
Table D-2 : Applicability matrix based on software criticality	90

Foreword

This document (prEN 16602-80:2016) has been prepared by Technical Committee CEN/CLC/TC 5 “Space”, the secretariat of which is held by DIN (Germany).

This document (prEN 16602-80:2016) originates from ECSS-Q-ST-80C Rev.1 DIR1.

This document is currently submitted to the CEN ENQUIRY.

This document has been developed to cover specifically space systems and will therefore have precedence over any EN covering the same scope but with a wider domain of applicability (e.g.: aerospace).

1

Scope

This Standard defines a set of software product assurance requirements to be used for the development and maintenance of software for space systems. Space systems include manned and unmanned spacecraft, launchers, payloads, experiments and their associated ground equipment and facilities. Software includes the software component of firmware.

This Standard also applies to the development or reuse of non-deliverable software which affects the quality of the deliverable product or service provided by a space system, if the service is implemented by software.

ECSS-Q-ST-80 interfaces with space engineering and management, which are addressed in the Engineering (-E) and Management (-M) branches of the ECSS System, and explains how they relate to the software product assurance processes.

This standard may be tailored for the specific characteristic and constraints of a space project in conformance with ECSS-S-ST-00.

Tailoring of this Standard to a specific business agreement or project, when software product assurance requirements are prepared, is also addressed in clause 4.3.

Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this ECSS Standard. For dated references, subsequent amendments to, or revision of any of these publications do not apply. However, parties to agreements based on this ECSS Standard are encouraged to investigate the possibility of applying the more recent editions of the normative documents indicated below. For undated references, the latest edition of the publication referred to applies.

EN reference	Reference in text	Title
EN 16601-00-01	ECSS-S-ST-00-01	ECSS system — Glossary of terms
EN 16003-40	ECSS-E-ST-40	Space engineering — Software general requirements
EN 16602-10	ECSS-Q-ST-10	Space product assurance – Product assurance management
EN 16602-10-04	ECSS-Q-ST-10-04	Space product assurance – Critical-item control
EN 16602-10-09	ECSS-Q-ST-10-09	Space product assurance – Nonconformance control system
EN 16602-20	ECSS-Q-ST-20	Space product assurance – Quality assurance
EN 16602-30	ECSS-Q-ST-30	Space product assurance – Dependability
EN 16602-40	ECSS-Q-ST-40	Space product assurance – Safety
EN 16601-10	ECSS-M-ST-10	Space project management – Project planning and implementation
EN 16601-10-01	ECSS-M-ST-10-01	Space project management – Organization and conduct of reviews
EN 16601-40	ECSS-M-ST-40	Space project management – Configuration and information management
EN 16601-80	ECSS-M-ST-80	Space project management – Risk management
	ISO/IEC 15504 Part 2:2003	Software engineering - Process assessment – Part 2: Performing an assessment - First Edition

3

Terms, definitions and abbreviated terms

3.1 Terms for other standards

For the purpose of this Standard, the terms and definitions from ECSS-ST-00-01 apply in particular for the term:

acceptance test

software product

NOTE The terms and definitions are common for the ECSS-E-ST-40 and ECSS-Q-ST-80 Standards.

3.2 Terms specific to the present standard

3.2.1 automatic code generation

generation of source code with a tool from a model

3.2.2 code coverage

percentage of the software that has been executed (covered) by the test suite

3.2.3 competent assessor

person who has demonstrated the necessary skills, competencies and experience to lead a process assessment in conformance with ISO/IEC 15504

NOTE Adapted from ISO/IEC 15504:1998, Part 9.

3.2.4 condition

boolean expression not containing boolean operators

3.2.5 configurable code

code (source code or executable code) that can be tailored by setting values of parameters

NOTE This definition covers in particular classes of configurable code obtained by the following configuration means:

- configuration based on the use of a compilation directive;

- configuration based on the use of a link directive;
- configuration performed through a parameter defined in a configuration file;
- configuration performed through data defined in a database with impact on the actually executable parts of the software (e.g. parameters defining branch structures that result in the non-execution of existing parts of the code).

3.2.6 COTS, OTS, MOTS software

for the purpose of this Standard, commercial-off-the-shelf, off-the-shelf and modified-off-the-shelf software for which evidence of use is available

3.2.7 critical software

software of criticality category A, B or C

NOTE See ECSS-Q-ST-80C, Annex D.1 – Software criticality categories.

3.2.8 deactivated code

code that, although incorporated through correct design and coding, is intended to execute in certain software product configurations only, or in none of them

[adapted from RTCA/DO-178B]

3.2.9 decision

boolean expression composed of conditions and zero or more boolean operators that are used in a control construct.

NOTE 1 For example: “if.....then else” or the “case” statement are control construct.

NOTE 2 A decision without a boolean operator is a condition.

NOTE 3 If a condition appears more than once in a decision, each occurrence is a distinct condition.

3.2.10 decision coverage

measure of the part of the program within which every point of entry and exit is invoked at least once and every decision has taken “true” and “false” values at least once.

NOTE Decision coverage includes, by definition, statement coverage.

3.2.11 existing software

any software developed outside the business agreement to which this Standard is applicable, including software from previous developments provided by the

supplier, software from previous developments provided by the customer, COTS, OTS and MOTS software, freeware and open source software

3.2.12 integration testing

testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them

[IEEE 610.12:1990]

3.2.13 logical model

implementation-independent model of software items used to analyse and document software requirements

3.2.14 margin philosophy

rationale for margins allocated to the performance parameters and computer resources of a development, and the way to manage these margins during the execution of the project

3.2.15 metric

defined measurement method and the measurement scale

NOTE 1 Metrics can be internal or external, and direct or indirect.

NOTE 2 Metrics include methods for categorising qualitative data.

[ISO/IEC 9126-1:2001]

3.2.16 migration

porting of a software product to a new environment

3.2.17 mission products

products and services delivered by the space system

NOTE For example: Communications services, science data.

3.2.18 modified condition and decision coverage

measure of the part of the program within which every point of entry and exit has been invoked at least once, every decision in the program has taken “true” and “false” values at least once, and each condition in a decision has been shown to independently affect that decision’s outcome

NOTE A condition is shown to independently affect a decision’s outcome by varying that condition while holding fixed all other possible conditions.

3.2.19 operational

for the purpose of this Standard, related to the software operation

NOTE It is not related to the spacecraft operation.

3.2.20 portability (a quality characteristic)

capability of software to be transferred from one environment to another

3.2.21 quality characteristics (software)

set of attributes of a software product by which its quality is described and evaluated

NOTE A software quality characteristic can have multiple levels of sub-characteristics.

3.2.22 quality model (software)

set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality

[ISO/IEC 9126-1:2001]

3.2.23 real-time

pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process

[IEEE 610.12:1990]

3.2.24 regression testing (software)

selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements

[IEEE 610.12:1990]

3.2.25 reusability

degree to which a software unit or other work product can be used in more than one computer program or software system

[IEEE 610.12:1990]

3.2.26 singular input

input corresponding to a singularity of the function

3.2.27 software

see “software product” in ECSS-S-ST-00-01

3.2.28 software component

part of a software system

NOTE 1 Software component is used as a general term.

NOTE 2 Components can be assembled and decomposed to form new components. In the production activities, components are implemented as units, tasks or programs, any of which can be configuration items. This usage of the term is more general than

in ANSI/IEEE parlance, which defines a component as a “basic part of a system or program”; in this Standard, components are not always “basic” as they can be decomposed.

3.2.29 software intensive system

space system in which the dominant part of the constituents are software elements

NOTE In such systems, subsystems consist mainly of software. For this type of system, the majority of interfaces are software-software interfaces.

3.2.30 software item

see “software product” in ECSS-S-ST-00-01

3.2.31 software observability

property of a system for which the values of status variables can be determined throughout observations of the output variables

3.2.32 software problem

condition of a software product that causes difficulty or uncertainty in the use of the software

[CMU/SEI-92-TR-022]

3.2.33 software product assurance

totality of activities, standards, controls and procedures in the lifetime of a software product which establishes confidence that the delivered software product, or software affecting the quality of the delivered product, conforms to customer requirements

3.2.34 software unit

separately compilable piece of source code

NOTE In this Standard no distinction is made between a software unit and a database; both are covered by the same requirements.

3.2.35 statement coverage

measure of the part of the program within which every executable source code statement has been invoked at least once.

3.2.36 stress test

test that evaluates a system or software component at or beyond its required capabilities

3.2.37 test case

set of test inputs, execution conditions and expected results developed for a particular objective such as to exercise a particular program path or to verify compliance with a specified requirement

3.2.38 test design

documentation specifying the details of the test approach for a software feature or combination of software features and identifying associated tests

3.2.39 test procedure

detailed instructions for the set up, operation and evaluation of the results for a given test

3.2.40 test script

file containing a set of commands or instructions written in native format (computer or tool processable) in order to automate the execution of one or a combination of test procedures (and the associated evaluation of the results)

3.2.41 unit test

test of individual software unit

3.2.42 unreachable code

code that cannot be executed due to design or coding error

3.2.43 usability (a quality characteristic)

capability of the software to be understood, learned, used and liked by the user, when used under specified conditions

3.2.44 validation

<software> process to confirm that the requirements baseline functions and performances are correctly and completely implemented in the final product

3.2.45 verification

<software> process to confirm that adequate specifications and inputs exist for any activity, and that the outputs of the activities are correct and consistent with the specifications and input

3.2.46 walk-through

static analysis technique in which a designer or programmer leads members of the development team and other interested parties through a software product, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems

[IEEE 1028-1997]

3.3 Abbreviated terms

For the purpose of this Standard and of ECSS-E-ST-40, the abbreviated terms from ECSS-S-ST-00-01 and the following apply:

For the definition of DRD acronyms see Annex A.

NOTE The abbreviated terms are common for the ECSS-E-ST-40 and ECSS-Q-ST-80 Standards.

Abbreviation	Meaning
AR	acceptance review NOTE The term SW-AR can be used for clarity to denote ARs that solely involve software products.
CDR	critical design review NOTE The term SW-CDR can be used for clarity to denote CDRs that solely involve software products.
CMMI	capability maturity model integration
COTS	commercial-off-the-shelf
CPU	central processing unit
DDF	design definition file
DDR	detailed design review
DJF	design justification file
DRD	document requirements definition
ECSS	European Cooperation for Space Standardization
eo	expected output
GS	ground segment
HMI	human machine interface
HSIA	hardware-software interaction analysis
HW	hardware
ICD	interface control document
INTRSA	international registration scheme for assessors
IRD	interface requirements document
ISO	International Organization for Standardization
ISV	independent software validation
ISVV	independent software verification and validation
MGT	management file
MF	maintenance file
MOTS	modified off-the-shelf
OBCP	on-board control procedure
OP	operational plan
ORR	operational readiness review
OTS	off-the-shelf
PAF	product assurance file
PDR	preliminary design review

NOTE The term SW-PDR can be used for clarity to denote PDRs that solely involve software products.

PRR preliminary requirement review

QR qualification review

NOTE The term SW-QR can be used for clarity to denote QRs that solely involve software products.

RB requirements baseline

SCAMPI standard CMMI appraisal method for process improvement

SDE software development environment

SOS software operation support

SPA software product assurance

SPAMR software product assurance milestone report

SPAP software product assurance plan

SPR software problem report

SRB software review board

SRR system requirements review

NOTE The term SW-SRR can be used for clarity to denote SRRs that solely involve software products.

SW software

SWE software engineering

TRR test readiness review

TS technical specification

4

Space system software product assurance principles

4.1 Introduction

The objectives of software product assurance are to provide adequate confidence to the customer and to the supplier that the developed or procured/reused software satisfies its requirements throughout the system lifetime. In particular, that the software is developed to perform properly and safely in its operational environment, meeting the quality objectives agreed for the project.

This Standard contributes to these objectives by defining the software product assurance requirements to be met in a particular space project. These requirements deal with quality management and framework, life cycle activities and process definition and quality characteristics of products.

One of the fundamental principles of this Standard is the customer-supplier relationship, assumed for all software developments. The organizational aspects of this are defined in ECSS-M-ST-10. The customer is, in the general case, the procurer of two strongly associated products: the hardware and the software components of a system, subsystem, set, equipment or assembly. The concept of the customer-supplier relationship is applied recursively, i.e. the customer can himself be a supplier to a higher level in the space system hierarchy.

The requirements of this Standard are applicable to the supplier, unless otherwise explicitly stated.

The supplier demonstrates compliance with the software product assurance requirements and provides the specified evidence of compliance.

To this end, the supplier specifies the software product assurance requirements for his/her suppliers, taking into account their responsibilities and the specific nature of their deliverables.

This Standard complements ECSS-E-ST-40 “Space engineering — Software general requirements”, with product assurance aspects, integrated in the space system software engineering processes as defined in ECSS-E-ST-40. Together the two standards specify all processes for space software development.

Figure 4-1 schematically presents the different Software processes addressed by the set of the ECSS standards.

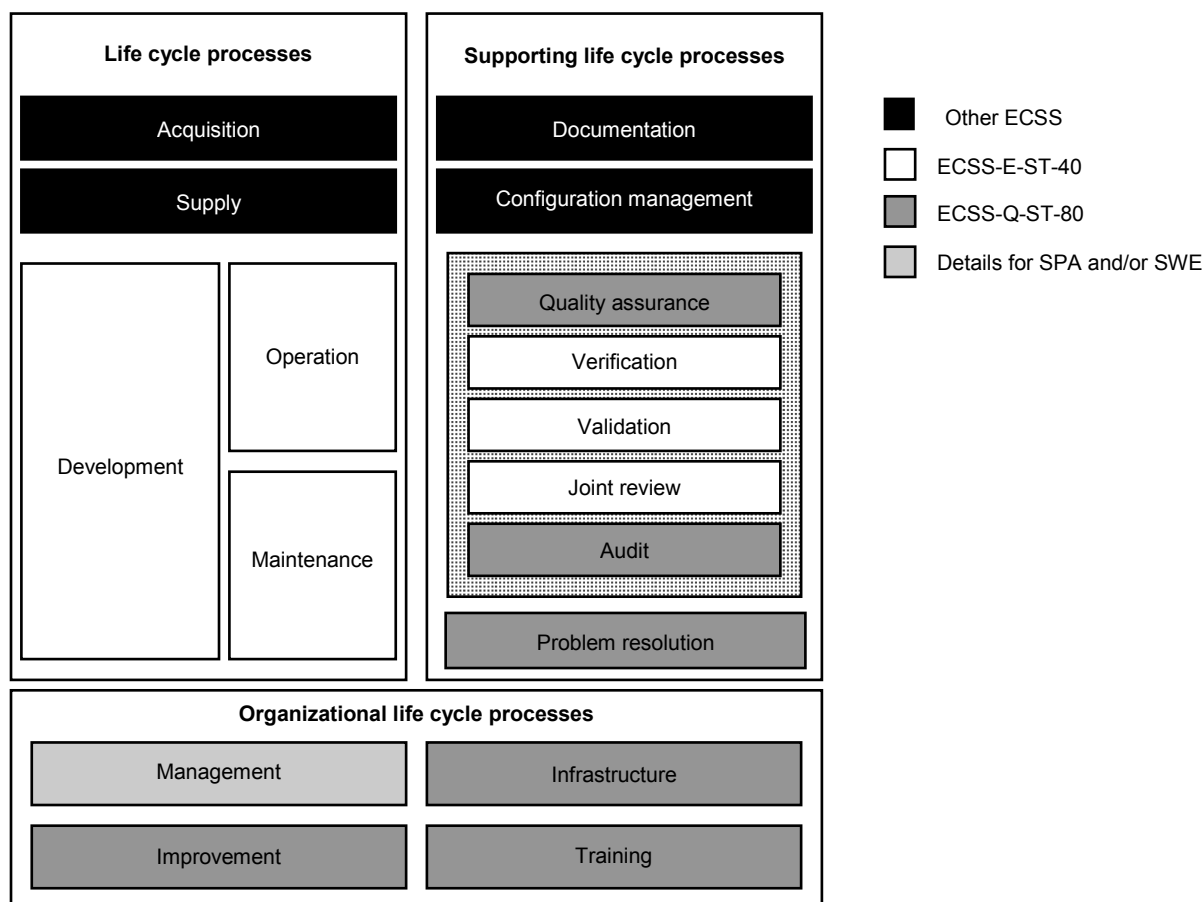


Figure 4-1: Software related processes in ECSS Standards

4.2 Organization of this Standard

This Standard is organized into three main parts:

- Software product assurance programme implementation
- Software process assurance
- Software product quality assurance.

The software documentation collecting the expected output of the ECSS-E-ST-40 and ECSS-Q-ST-80 requirements is summarized in Annex A.

Annex B and Annex C specify the DRDs (document requirements definitions) of the software product assurance documents (SPAP and SPAMR). The DRDs of other software engineering and management documents are included in ECSS-E-ST-40 and ECSS-M-ST-40.

In the preparation of this Standard the ISO/IEC 12207 standard has been used extensively, providing a common internationally recognized framework for the terminology and software life cycle processes description.

The organization of this Standard is reflected in detail in Figure 4-2.

Software product assurance programme implementation	
5.1 Organization and responsibility	5.5 Procurement
5.2 Software product assurance programme management	5.6 Tools and supporting environment
5.3 Risk management and critical item control	5.7 Assessment and improvement process
5.4 Supplier selection and control	

Software process assurance
6.1 Software development life cycle
6.2 Requirements applicable to all software engineering processes
6.3 Requirements applicable to individual software engineering processes or activities

Software product quality assurance
7.1 Product quality objectives and metrication
7.2 Product quality requirements
7.3 Software intended for reuse
7.4 Standard ground hardware and services for operational system
7.5 Firmware

Figure 4-2: Structure of this Standard

Each requirement of this Standard is identified by a hierarchical number, plus a letter if necessary (e.g. 5.3.1.5, bullet a). For each requirement, the associated output is given in the “Expected Output” section. When several outputs are expected, they are identified by a letter (e.g. “a”, “b”, etc.). With each output, the destination file of the output is indicated in brackets, together with the corresponding document DRD (after a comma) and review(s) (after a semicolon). For example: “[PAF, SPAP; SRR]” denotes an output contained in the Software Product Assurance Plan, part of the Product Assurance File, and required at SRR. When no DRD is defined for an Expected Output, and/or the Expected Output is not to be provided at any specific milestone review, then the corresponding sections of that Expected Output are replaced by dashes (e.g. “[PAF, -; -]”).

This standards details for the Software Product Assurance aspects some of the general requirements already addressed by the ECSS Management, Product Assurance and Quality Assurance standards.

4.3 Tailoring of this Standard

The general information and requirements for the selection and tailoring of applicable standards are defined in ECSS-S-ST-00.

There are several drivers for tailoring, such as dependability and safety aspects, software development constraints, product quality objectives and business objectives.

Tailoring for dependability and safety aspects is based on the selection of requirements related to the verification, validation and levels of proofs demanded by the criticality of the software. Annex D contains a tailoring of this Standard based on software criticality.

Tailoring for software development constraints takes into account the special characteristics of the software being developed, and of the development environment. The type of software development (e.g. database or real-time) and the target system (e.g. embedded processor, host system, programmable devices, or application-specific integrated circuits) are also taken into account (see Annex S of ECSS-E-ST-40). Specific requirements for verification, review and inspection are imposed, for example, when full validation on the target computer is not feasible or where performance goals are difficult to achieve.

Tailoring for product quality and business objectives is done by selecting requirements on quality of the product as explained in clause 7 of this Standard based on the quality objectives for the product specified by the customer.

5

Software product assurance programme implementation

5.1 Organization and responsibility

5.1.1 Organization

- a. The supplier shall ensure that an organizational structure is defined for software development, and that individuals have defined tasks and responsibilities.

5.1.2 Responsibility and authority

5.1.2.1

- a. The responsibility, the authority and the interrelation of personnel who manage, perform and verify work affecting software quality shall be defined and documented.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.2.2

- a. The responsibilities and the interfaces of each organisation, either external or internal, involved in a project shall be defined and documented.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.2.3

- a. The delegation of software product assurance tasks by a supplier to a lower level supplier shall be done in a documented and controlled way, with the supplier retaining the responsibility towards the customer.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.3 Resources

5.1.3.1

- a. The supplier shall provide adequate resources to perform the required software product assurance tasks.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.3.2

- a. Reviews and audits of processes and of products shall be carried out by personnel not directly involved in the work being performed.

5.1.4 Software product assurance manager/engineer

5.1.4.1

- a. The supplier shall identify the personnel responsible for software product assurance for the project (SW PA manager/engineer).

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR].

5.1.4.2

- a. The software product assurance manager/engineer shall
 1. report to the project manager (through the project product assurance manager, if any);
 2. have organisational authority and independence to propose and maintain a software product assurance programme in accordance with the project software product assurance requirements;
 3. have unimpeded access to higher management as necessary to fulfil his/her duties.

5.1.5 Training

5.1.5.1

- a. The supplier shall review the project requirements to establish and make timely provision for acquiring or developing the resources and skills for the management and technical staff.

EXPECTED OUTPUT: Training plan [MGT, -; SRR].

5.1.5.2

- a. The supplier shall maintain training records.

EXPECTED OUTPUT: Records of training and experience [PAF, -; -].

5.1.5.3

- a. The supplier shall ensure that the right composition and categories of appropriately trained personnel are available for the planned activities and tasks in a timely manner.

5.1.5.4

- a. The supplier shall determine the training subjects based on the specific tools, techniques, methodologies and computer resources to be used in the development and management of the software product.

NOTE Personnel can undergo training to acquire skills and knowledge relevant to the specific field with which the software is to deal.

5.2 Software product assurance programme management

5.2.1 Software product assurance planning and control

5.2.1.1

- a. The supplier shall develop a software product assurance plan in response to the software product assurance requirements in conformance with DRD in annex B.
- b. The software product assurance plan shall be either a standalone document or a section of the supplier overall product assurance plan.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR, PDR].

5.2.1.2

- a. Any internal manuals, standards or procedures referred to by the software product assurance plan shall become an integral part of the supplier's software product assurance programme.

5.2.1.3

- a. The software product assurance plan shall be revisited and updated as needed at each milestone to ensure that the activities to be undertaken in the following phase are fully defined.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; CDR, QR, AR, ORR].

5.2.1.4

- a. Before acceptance review, the supplier shall either supplement the software product assurance plan to specify the quality measures related

to the operations and maintenance processes, or issue a specific software product assurance plan.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; AR].

5.2.1.5

- a. The supplier shall provide with the software product assurance plan a compliance matrix documenting conformance with the individual software product assurance requirements applicable for the project or business agreement.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR, PDR].

- b. For each software product assurance requirement, the compliance matrix shall provide a reference to the document where the expected output of that requirement is located.

NOTE For compliance with the required DRDs a general statement of compliance is acceptable.

EXPECTED OUTPUT: Software product assurance plan [PAF, SPAP; SRR, PDR].

5.2.2 Software product assurance reporting

5.2.2.1

- a. The supplier shall report on a regular basis on the status of the software product assurance programme implementation, if appropriate as part of the overall product assurance reporting of the project.

EXPECTED OUTPUT: Software product assurance reports [PAF, -; -].

5.2.2.2

- a. The software product assurance report shall include:
1. an assessment of the current quality of the product and processes, based on measured properties, with reference to the metrication as defined in the software product assurance plan;
 2. verifications undertaken;
 3. problems detected;
 4. problems resolved.

EXPECTED OUTPUT: Software product assurance reports [PAF, -; -].

5.2.2.3

- a. The supplier shall deliver at each milestone review a software product assurance milestone report, covering the software product assurance activities performed during the past project phases.

EXPECTED OUTPUT: Software product assurance milestone report [PAF, SPAMR; SRR, PDR, CDR, QR, AR, ORR].

5.2.3 Audits

- a. For software audits, ECSS-Q-ST-10 clause 5.2.3 shall apply.

EXPECTED OUTPUT: Audit plan and schedule [PAF, -; SRR].

5.2.4 Alerts

- a. For software alerts, ECSS-Q-ST-10 clause 5.2.9 shall apply.

EXPECTED OUTPUT: The following outputs are expected:

- a. Preliminary alert information [PAF, -; -];
- b. Alert information [PAF, -; -].

5.2.5 Software problems

5.2.5.1

- a. The supplier shall define and implement procedures for the logging, analysis and correction of all software problems encountered during software development.

EXPECTED OUTPUT: Software problem reporting procedures [PAF, -; PDR].

5.2.5.2

- a. The software problem report shall contain the following information:

1. identification of the software item;
2. description of the problem;
3. recommended solution;
4. final disposition;
5. modifications implemented (e.g. documents, code, and tools);
6. tests re-executed.

EXPECTED OUTPUT: Software problem reporting procedures [PAF, -; PDR].

5.2.5.3

- a. The procedures for software problems shall define the interface with the nonconformance system (i.e. the circumstances under which a problem qualifies as a nonconformance).

EXPECTED OUTPUT: Software problem reporting procedures [PAF, -; PDR].

5.2.5.4

- a. The supplier shall ensure the correct application of problem reporting procedures.